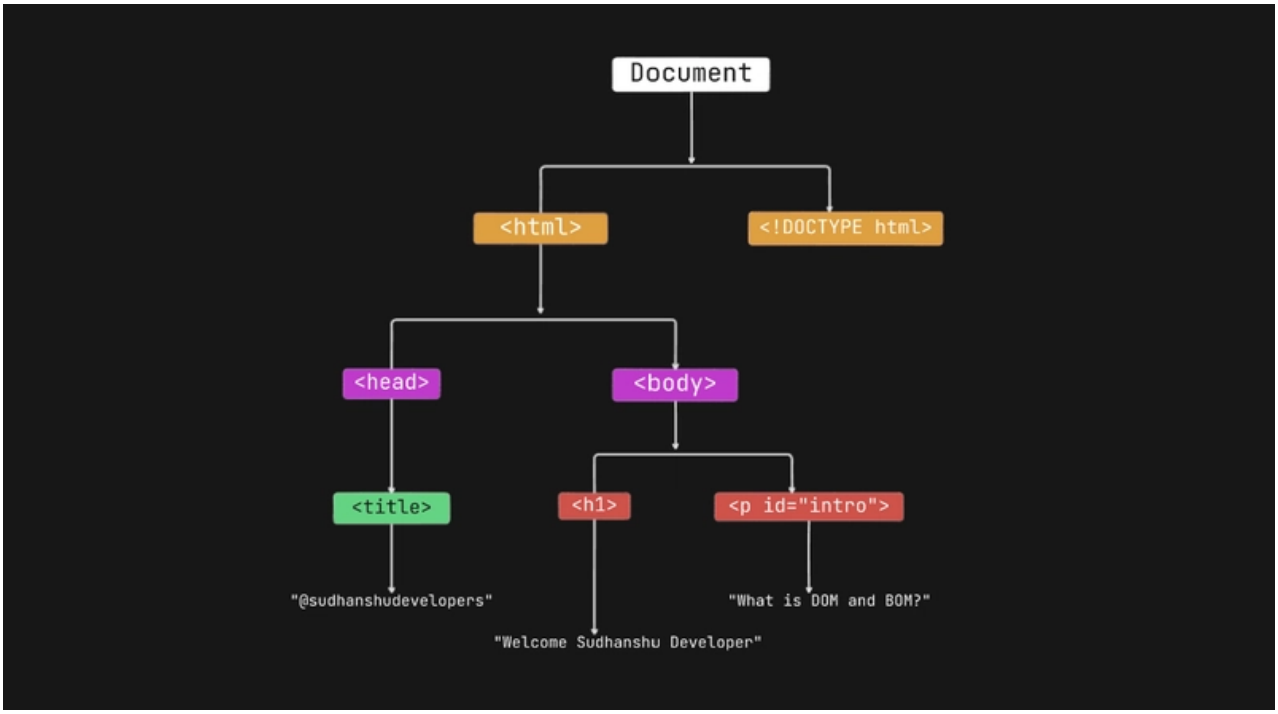
- Text Nodes: The text inside tags (e.g., “What is DOM and BOM?”) becomes text nodes.
- Attribute Nodes: Attributes like id or class are linked to their respective elements.

For example, consider this HTML:

```
<!DOCTYPE html>
<html>
  <head>
    <title>@sudhanshudevelopers</title>
  </head>
  <body>
    <h1>Welcome Sudhanshu Developer</h1>
    <p id="intro">What is DOM and BOM?</p>
  </body>
</html>
```

Graphical DOM Tree representation



HTML DOM Methods

The DOM provides a rich set of methods to interact with this tree. Here are some key ones with examples:

1.Accessing Elements

- `document.getElementById(id)`: Finds an element by its ID.
- `document.querySelector(selector)`: Returns the first element matching a CSS selector.

```
const intro = document.getElementById('intro');
console.log(intro.textContent); // "This is a test."

const heading = document.querySelector('h1');
console.log(heading.textContent); // "Welcome"
```

2.Modifying Elements

`element.innerHTML`: Sets or gets the HTML inside an element.
`element.textContent`: Sets or gets the text content.

```
intro.innerHTML = '<strong>Updated text!</strong>';
heading.textContent = 'Hello, Developers!';
```

3.Creating and Adding Elements

`document.createElement(tagName)`: Creates a new element.
`parent.appendChild(child)`: Adds a child to a parent.

```
const newPara = document.createElement('p');
newPara.textContent = 'I’m new here!';
document.body.appendChild(newPara);
```

4.Removing Elements

`element.remove()`: Deletes an element from the DOM.

```
intro.remove(); // Goodbye, intro paragraph!
```

The Browser Object Model (BOM)

While the DOM deals with the document’s structure, the **Browser Object Model** (BOM) provides access to the browser itself. The BOM isn’t standardized like the DOM, but it’s widely supported and includes objects like window, navigator, location, and more.

1.`window`Object

Represents the browser window and serves as the global object in JavaScript. Includes properties like `innerWidth` and methods like `alert()`.

```
console.log(window.innerWidth); // Browser window width in pixels
window.alert('Hello from the BOM!');
```

2.`navigator` Object

Provides info about the user's browser and system.

```
console.log(navigator.userAgent); // Browser info, e.g., "Mozilla/5.0..."
console.log(navigator.language); // User's preferred language, e.g., "en-US"
```

3.`location`Object

Manages the browser's URL and navigation.

```
console.log(location.href); // Current URL
// location.href = 'https://example.com'; // Redirects to a new page
```

4.history Object

Controls browser history.

```
history.back(); // Go to previous page
history.forward(); // Go to next page
```

5.screen Object

Provides info about the user's screen.

```
console.log(screen.width); // Screen width in pixels
console.log(screen.height); // Screen height in pixels
```

Practical BOM Example: A Simple Redirect Button

```
<button onclick="redirect()">Go to Google</button>

<script>
  function redirect() {
    if (confirm('Leave this page?')) {
      location.href = 'https://www.google.com';
    }
  }
</script>
```

This uses the `window.confirm()` method (BOM) to ask for confirmation and `location.href` to redirect.

DOM vs. BOM: What's the Difference?

- DOM: Focuses on the document (HTML structure). Think elements, attributes, and content.
- BOM: Focuses on the browser. Think windows, navigation, and screen info.

- Overlap: The `window.document` object is where they meet—`document`is part of the DOM, but it's accessed via the BOM's `window`.

Putting It All Together: A Real-World Example

Let's combine DOM and BOM to create a dynamic page that greets the user based on their browser and screen size:

```
<!DOCTYPE html>
<html>
  <body>
    <h1 id="greeting"></h1>
    <p id="info"></p>

    <script>
      // DOM: Access elements
      const greeting = document.getElementById('greeting');
      const info = document.getElementById('info');

      // BOM: Get browser and screen info
      const browser = navigator.userAgent.includes('Chrome') ? 'Chrome' : 'Another
Browser';
      const screenSize = `${screen.width}x${screen.height}`;

      // DOM: Update the page
      greeting.textContent = `Hello, ${browser} User!`;
      info.textContent = `Your screen resolution is ${screenSize}.`;
    </script>
  </body>
</html>
```

Output (varies by user):

"Hello, Chrome User!"
"Your screen resolution is 1920x1080."

Conclusion

The HTML DOM and BOM are essential tools in a JavaScript developer's toolkit. The DOM lets you manipulate the page's structure and content, while the BOM gives you control over the browser environment. Together, they enable rich, interactive web experiences. Try experimenting with these examples—add some DOM manipulation or BOM features to your next project and see the magic happen!

Top comments (0)

Subscribe